



Table of Contents

Overview	3
Purpose of This Guide	3
Prerequisites	3
LEAP Syntax	4
Definitions	4
System Resources	4
Supported Requests	4
More Information	4
Set the LEAP Version Number	5
Keep the LEAP Connection Open	6
Sample Commands	6
Discover Resources	6
Discover the Root Area of a Lutron System	7
Discover Child Areas of the Root Area	7
Continue Discovery of Areas	8
Discover Devices in Leaf Areas	10
Discover Buttons on Control Stations	12
Discover Lighting, Shade, and HVAC Zones	13
Discover Area Scenes	16
Controlling Resources	17
Changing a Dimmed Lighting Zone's Level	17
Changing a Shade Zone's Level	18
Changing an HVAC Zone's Setpoint	19
Calling an Area Scene	20
Monitoring Resources	20
Monitor Zone Changes	20
Monitor Area Changes	22
Next Steps	22
Revision History	23

Overview

Lutron's Extensible Application Protocol (LEAP) enables third-party equipment, such as touchscreens, universal remotes, and software applications, to discover, control, and monitor devices in a Lutron System.

Purpose of This Guide

The LEAP Quick Start Guide is broken into three distinct parts:

- will walk you through the prerequisites needed to understand and implement integration with a Lutron system and will also explain how to gain access to Lutron's Developer Portal as an approved Lutron integrator.
- will explain the process for generating and signing the certificates that are required to initiate the connection with a Lutron system, as well as the steps for performing the connection.
- (this document) will provide basic commands to help you become familiar with using the LEAP protocol to control the Lutron system.

This guide is designed to empower the user to quickly access a Lutron system via their approved API and build the user's confidence in doing so. This is not intended to be a comprehensive guide for developing a complete API. For an exhaustive list of available commands, refer to Lutron's complete API Protocol Guide for the relevant system, which is available on Lutron's Developer Portal.

The following Lutron commercial and residential systems are covered by the LEAP Quick Start Guide:

- Athena
- HomeWorks QSX
- RadioRA 3
- myRoom XC

For LEAP integration with Lutron's commercial Quantum and Vive systems, or with Lutron's residential Caseta and RA2 Select systems, email workswith@lutron.com for more information.

Prerequisites

This guide assumes a prior knowledge of the following:

1. Familiarity with OpenSSL for the creation and cross signing of certificates (or other methods of doing so)
2. How to establish TLS connections and use raw sockets (HTTP based applications such as Postman will not work)
3. How to create and parse JSON messages
4. How to process standard HTTP response codes

LEAP Syntax

LEAP is built on the RESTful protocol, and commands are written to and received from the Lutron system in JSON format.

Definitions

Throughout the rest of Part 3 of this guide, the following terms will be used:

- Processor: the Lutron system processor
- Client: the integration device
- *href*: the reference ID of a resource in the Lutron system

System Resources

Every resource within the Lutron system's database is assigned a reference ID, referred to in LEAP as an *href*. The *hrefs* for the connected system can be discovered entirely through LEAP, which will be explained in a later section of this guide.

Supported Requests

LEAP supports the following types of requests from clients, and the processor will send a corresponding, synchronous response for each of the below requests.

Request/Response Pairs	
CreateRequest	CreateResponse
ReadRequest	ReadResponse
UpdateRequest	UpdateResponse
DeleteRequest	DeleteResponse
SubscribeRequest	SubscribeResponse
UnsubscribeRequest	UnsubscribeResponse

Note the following for SubscribeRequests:

- In addition to synchronous response confirming subscription, the client will also receive asynchronous ReadResponses from the processor whenever the resource subscribed to changes.
- If the subscribed resource is deleted, the client will receive an asynchronous DeleteResponse from the processor.
- A client may only subscribe to a unique resource once. Multiple subscriptions to the same resource from the same client are not supported. If a client does subscribe to the same resource twice, the original subscription will be discarded and replaced by the most recent subscription.

More Information

The information provided in the above sections gives enough background to get started using LEAP. The complete API Protocol Guide provides more detailed information regarding LEAP syntax, which should be considered while developing your full integration.

Set the LEAP Version Number

As LEAP is developed, new major versions may be released which could affect compatibility with clients using older versions of LEAP. By defining the desired major version, the processor will be able to negotiate requests and responses with the client.



Keep the LEAP Connection Open

LEAP only allows for 10 concurrent LEAP connections to a processor. The processor will automatically disconnect the connection if a client does not send a command over the course of three minutes. The below command can be used to send a ping to the processor, keeping the LEAP connection open.

```
{
    "CommuniqueType": "ReadRequest",
    "Header": {
        "URL": "/server/status/ping"
    }
}

{
    "CommuniqueType": "ReadResponse",
    "Header": {
        "MessageBodyType": "OnePingResponse",
        "StatusCode": "200 OK",
        "Url": "/server/status/ping"
    },
    "Body": {
        "PingResponse": {
            "LEAPVersion": 3.231
        }
    }
}
```

Sample Commands

The following subsections provide sample code that can be used for testing API development. Note that certain objects/parameters in the sample code are database-specific references and will need to be updated using the *refs* from the Lutron system to which you are connecting.

Discover Resources

Before monitoring or controlling system resources, their *refs* must first be discovered. LEAP commands can be used to discover these *refs*, and this is the best practice for production-level integrations on live projects. For development purposes, an Integration Report that lists available *refs* can also be generated from the Lutron Designer software.

The code samples below will walk through the following process:

1. Discover the Lutron system's area hierarchy
 - a. Discover the root area for the connected Lutron system
 - b. Discover child areas of the root area
 - c. Continue discovering child areas of each previously discovered areas until all areas have been found.
 - i. The lowest level of areas within the area hierarchy are called *leaf areas* and will contain the parameter “*IsLeaf*”: *true*.
2. Discover Devices within each area
 - a. Discover buttons on control stations
3. Discover lighting and shade zones
4. Discover area scenes

A system's area hierarchy is based on how the database was organized during initial programming. Your integration may need to repeat step 1c above multiple times for each new area that is discovered before all *leaf areas* are discovered.

Be advised that certain parameters for any given resource within the Lutron system may change as the system is used and updated. For example, users of the Lutron App are empowered to make changes to the programming of their system, including changing area names, scene names, and values for resources within area scenes. It may be necessary for your integration to run through the discovery process regularly to determine whether any resources have changed.

Discover the Root Area of a Lutron System

Send a ReadRequest with a URL of /area/rootarea.

<pre>{ "CommuniqueType": "ReadRequest", "Header": { "Url": "/area/rootarea" } }</pre>	<pre>{ "CommuniqueType": "ReadResponse", "Header": { "MessageBodyType": "OneAreaDefinition", "StatusCode": "200 OK", "Url": "/area/rootarea" }, "Body": { "Area": { "href": "/area/3", "XID": "1n4bEBBDG21n7UMriHSdA", "Name": "LEAP Sample Project", "IsLeaf": false } } }</pre>

Discover Child Areas of the Root Area

Send a ReadRequest calling a /childarea/summary on the *href* for the Root Area. If your system's Root Area *href* is not /area/3, remember to update the code to reflect the *href* value for your system.

<pre>{ "CommuniqueType": "ReadRequest", "Header": { "Url": "/area/3/childarea/summary" } }</pre>	<pre>{ "CommuniqueType": "ReadResponse", "Header": { "MessageBodyType": "MultipleAreaSummaryDefinition", "StatusCode": "200 OK", "Url": "/area/3/childarea/summary" }, "Body": { "AreaSummaries": [{ "Area": { "href": "/area/3", "XID": "1n4bEBBDG21n7UMriHSdA", "Name": "LEAP Sample Project" } }] } }</pre>

```
"href":"/area/2072",  
"Name":"Floor 1",  
"Parent":{
```

```

        "SortOrder":1,
        "IsLeaf":false
    }
]
}
}
}
```

<pre>{ "CommuniqueType": "ReadRequest", "Header": { "Url": "/area/127/childarea/summary" } }</pre>	<pre>{ "CommuniqueType": "ReadResponse", "Header": { "MessageBodyType": "MultipleAreaSummaryDefinition", "StatusCode": "200 OK", "Url": "/area/127/childarea/summary" }, "Body": { "AreaSummaries": [{ "href": "/area/602", "Name": "Open Office", "Parent": { "href": "/area/127" }, "SortOrder": 0, "IsLeaf": true }, { "href": "/area/616", "Name": "Private Office", "Parent": { "href": "/area/127" }, "SortOrder": 1, "IsLeaf": true }] } }</pre>
--	---

Discover Devices in Leaf Areas

With the system's area hierarchy fully discovered, it is now possible to discover the devices that live within each leaf area. The below sample code discovers the devices within the Private Office (*href: /area/616*), which include a daylight sensor (*href: /device/1252*), an occupancy sensor (*href: /device/1298*), and a wall control (*href: /device/1122*).

Due to the length of the lines, the response is provided below the request for this example.

```
{  
    "CommuniqueType": "ReadRequest",  
    "Header": {  
        "Url": "/area/616/associatedcontrolstation"  
    }  
}
```

```
{  
    "CommuniqueType": "ReadResponse",  
    "Header": {  
        "MessageBodyType": "MultipleControlStationDefinition",  
        "StatusCode": "200 OK",  
        "Url": "/area/616/associatedcontrolstation"  
    },  
    "Body": {  
        "ControlStations": [  
            {  
                "href": "/controlstation/1120",  
                "Name": "Control Station 001",  
                "AssociatedArea": {  
                    "href": "/area/616"  
                },  
                "SortOrder": 0,  
                "AssociatedGangedDevices": [  
                    {  
                        "Device": {  
                            "DeviceType": "SeeTouchKeypad",  
                            "href": "/device/1122",  
                            "AddressedState": "Unaddressed"  
                        },  
                        "GangPosition": 0  
                    }  
                ]  
            }  
        ]  
    }  
}
```

```
        },
        {
            "href":"/controlstation/1250",
            "Name":"RF Daylight Sensor 001",
            "AssociatedArea":{
                "href":"/area/616"
            },
            "SortOrder":1,
            "AssociatedGangedDevices":[
                {
                    "Device":{
                        "DeviceType":"RPSDaylightSensor",
                        "href":"/device/1252",
                        "AddressedState":"Unaddressed"
                    },
                    "GangPosition":0
                }
            ]
        },
        {
            "href":"/controlstation/1296",
            "Name":"RF Occupancy Sensor 001",
            "AssociatedArea":{
                "href":"/area/616"
            },
            "SortOrder":2,
            "AssociatedGangedDevices":[
                {
                    "Device":{
                        "DeviceType":"RPSCeilingMountedOccupancySensor",
                        "href":"/device/1298",
                        "AddressedState":"Unaddressed"
                    },
                    "GangPosition":0
                }
            ]
        }
    }
}
```

Discover Buttons on Control Stations

The previous example provided some details about a wall control in the Private Office, but it did not list every component. The device in question is a keypad that has multiple buttons, all of which are programmed in the Lutron system to trigger different commands. For a client to mimic button presses on this keypad, the *hrefs* for those buttons must be discovered by requesting the expanded button group information, as per the below code.

Due to the length of the lines, the response is provided below the request for this example. For brevity, the response shown only provides information for a single button on the example keypad. This particular control has a total of seven buttons, each of which would have the same level of detail provided in the response message.

```
{  
    "CommuniqueType": "ReadRequest",  
    "Header": {  
        "Url": "/device/1122/buttongroup/expanded"  
    }  
}
```

```
{  
    "CommuniqueType": "ReadResponse",  
    "Header": {  
        "MessageBodyType": "MultipleButtonGroupExpandedDefinition",  
        "StatusCode": "200 OK",  
        "Url": "/device/1122/buttongroup/expanded"  
    },  
    "Body": {  
        "ButtonGroupsExpanded": [  
            {  
                "href": "/buttongroup/1134",  
                "Parent": {  
                    "href": "/device/1122"  
                },  
                "SortOrder": 0,  
                "ProgrammingType": "Freeform",  
                "Buttons": [  
                    {  
                        "href": "/button/1136",  
                        "ButtonNumber": 1,  
                        "ProgrammingModel": {  
                            "href":  
                                "/programmingmodel/1163",  
                            "Name": "Lutron  
                        }  
                    }  
                ]  
            }  
        ]  
    }  
}
```

```

    "ProgrammingModelType":"SingleActionProgrammingModel"
},
"Parent":{

    "href":"/buttongroup/1134"
},
"Name":"Button 1",
"Engraving":{

    "Text":""


},
"AssociatedLED":{

    "href":"/led/1129"
}
},

```

Discover Lighting, Shade, and HVAC Zones

Similar to devices, to discover lighting, shade, and HVAC zones for each area in a Lutron system, use `/associatedzone` on an area `href`. The response will include information for all three types of zones (lighting, shades, and HVAC) if they are present in the called area.

Note: HVAC control is only available in HomeWorks QSX and myRoom XC systems.

<pre>{ "CommuniqueType":"ReadRequest", "Header":{ "Url":"/area/616/associatedzone" } }</pre>	<pre>{ "CommuniqueType":"ReadResponse", "Header":{ "MessageBodyType":"MultipleZoneDefinition", "StatusCode":"200 OK", "Url":"/area/616/associatedzone" }, "Body":{ "Zones":[{ "href":"/zone/1698", "XID":"7N4cYSM6TEquN0Oun77VmA", "Name":"a", "ControlType":"Dimmed", "Category":{ "Type":"", "IsLight":true }, "AssociatedArea":{ "href":"/area/616" }, "SortOrder":1 }] } }</pre>

```

        },
        {
            "href": "/zone/1422",
            "Name": "Shade Group 1",
            "ControlType": "Shade",
            "Category": {
                "Type": "",
                "IsLight": false
            },
            "AssociatedArea": {
                "href": "/area/616"
            },
            "SortOrder": 0
        }
    ]
}
}

```

The below is an example of a response showing an HVAC zone.

```

{
    "CommuniqueType": "ReadResponse",
    "Header": {
        "MessageBodyType": "MultipleZoneDefinition",
        "StatusCode": "200 OK",
        "Url": "/area/1102/associatedzone"
    },
    "Body": {
        "Zones": [
            {
                "href": "/zone/1675",
                "Name": "HVAC Zone 001",
                "ControlType": "DualSetPointHVAC",
                "Category": {
                    "Type": "",
                    "IsLight": false
                },
                "DualSetPointHVACProperties": {
                    "TemperatureScales": [
                        "Fahrenheit"
                    ],
                    "FanModes": [

```

```
        "Auto",
        "On",
        "High",
        "Medium",
        "Low"
    ],
    "OperatingModes": [
        "Off",
        "Heat",
        "Cool",
        "Auto",
        "EmergencyHeat"
    ],
    "HeatingSetPointRange": {
        "F": {
            "Min": 50,
            "Max": 80,
            "Step": 1
        }
    },
    "CoolingSetPointRange": {
        "F": {
            "Min": 60,
            "Max": 90,
            "Step": 1
        }
    },
    "HeatingCoolingDelta": {
        "F": {
            "Min": 3
        }
    },
    "Schedule": {
        "ScheduleType": "WeeklySchedule",
        "WeeklySchedule": {
            "href": "/weeklyschedule/1675",
            "Sunday": {
                "href": "/dailyschedule/1715"
            },
            "Monday": {
                "href": "/dailyschedule/1720"
            },
            "Tuesday": {
                "href": "/dailyschedule/1720"
            }
        }
    }
}
```

```
        "Wednesday": {
            "href": "/dailyschedule/1720"
        },
        "Thursday": {
            "href": "/dailyschedule/1720"
        },
        "Friday": {
            "href": "/dailyschedule/1720"
        },
        "Saturday": {
            "href": "/dailyschedule/1715"
        }
    }
}
]
```

Discover Area Scenes

Each area in a Lutron system can have up to 16 unique area scenes, plus an “All Off” scene. Scenes are initially defined in the Lutron programming software and are used to control any combination of resources that exist within that area. Call /areascene on an area href to discover that area’s scenes.

```
{  
    "CommuniquType": "ReadRequest",  
    "Header": {  
        "URL": "/area/616/areascene"  
    }  
}  
  
{  
    "CommuniquType": "ReadResponse",  
    "Header": {  
        "MessageBodyType": "MultipleAreaSceneDefinition",  
        "StatusCode": "200 OK",  
        "Url": "/area/616/areascene"  
    },  
    "Body": {  
        "AreaScenes": [  
            {  
                "href": "/areascene/620",  
                "Name": "Off",  
                "Parent": {  
                    "href": "/area/616"  
                },  
                "Preset": {  
                    "href": "/preset/620"  
                }  
            }  
        ]  
    }  
}
```

```
        },
        "SortOrder":0
    },
    {
        "href":"/areascene/621",
        "Name":"Full On",
        "Parent":{
            "href":"/area/616"
        },
        "Preset":{
            "href":"/preset/621"
        },
        "SortOrder":1
    },
    {
        "href":"/areascene/622",
        "Name":"75%",
        "Parent":{
            "href":"/area/616"
        },
        "Preset":{
            "href":"/preset/622"
        },
        "SortOrder":2
    },
    //additional scenes omitted for brevity...
]
}
}
```

Controlling Resources

The next few examples will utilize CreateRequests instead of ReadRequests, as instead of reading information from the processor we will be telling the processor to do something with a system resource.

Changing a Dimmed Lighting Zone's Level

LEAP is able to change multiple parameters of any given lighting zone. The parameters available will vary depending on the zone's ControlType parameter. The below example can be used to control zones with ControlType equal to "Dimmed". Note that there are other parameters for "Dimmed" zones that have been omitted from sample below, such as FadeTime and DelayTime. Refer to the full API Protocol Guide for your system type for a complete list of available parameters for each ControlType.

<pre>{ "CommuniqueType": "CreateRequest", "Header": { "URL": "/zone/1698/commandprocessor" }, "Body": { "Command": { "CommandType": "GoToDimmedLevel", "DimmedLevelParameters": { "Level": 50 } } } }</pre>	<pre>{ "CommuniqueType": "CreateResponse", "Header": { "MessageBodyType": "OneZoneStatus", "StatusCode": "201 Created", "Url": "/zone/1698/commandprocessor" }, "Body": { "ZoneStatus": { "href": "/zone/1698/status", "Level": 50, "Zone": { "href": "/zone/1698" }, "StatusAccuracy": "Good", "Availability": "Available" } } }</pre>
---	---

Changing a Shade Zone's Level

The request to adjust a shade zone's level is very similar to the request for a "Dimmed" lighting zone.

<pre>{ "CommuniqueType": "CreateRequest", "Header": { "URL": "/zone/1422/commandprocessor" }, "Body": { "Command": { "CommandType": "GoToShadeLevel", "ShadeLevelParameters": { "Level": 50.00 } } } }</pre>	<pre>{ "CommuniqueType": "CreateResponse", "Header": { "MessageBodyType": "OneZoneStatus", "StatusCode": "201 Created", "Url": "/zone/1422/commandprocessor" }, "Body": { "ZoneStatus": { "href": "/zone/1422/status", "Level": 50, "Zone": { "href": "/zone/1422" }, "StatusAccuracy": "Good", "Availability": "Available" } } }</pre>
--	---

Changing an HVAC Zone's Setpoint

To adjust an HVAC zone's setpoint, an UpdateRequest is used instead of a CreateRequest. The below request is for a Dual Setpoint ControlType. Refer to the full API protocols for information on updating other HVAC ControlTypes.

<pre>{ "CommuniqueType": "UpdateRequest", "Header": { "URL": "/zone/1675/status" }, "Body": { "ZoneStatus": { "DualSetPointHvacStatus": { "HeatingSetPoint": { "F": 70 }, "CoolingSetPoint": { "F": 76 }, "OperatingMode": "Cool", "FanMode": "On" } } } }</pre>	<pre>{ "CommuniqueType": "UpdateResponse", "Header": { "MessageBodyType": "OneZoneStatus", "StatusCode": "200 OK", "Url": "/zone/1675/status" }, "Body": { "ZoneStatus": { "href": "/zone/1675/status", "Zone": { "href": "/zone/1675" }, "StatusAccuracy": "Good", "DualSetPointHVACStatus": { "CurrentTemperature": { "F": 72 }, "HeatingSetPoint": { "F": 70 }, "CoolingSetPoint": { "F": 76 }, "OperatingMode": "Cool", "OperatingStatuses": ["HeatLast"], "FanMode": "On", "FanStatus": "Unknown", "ScheduleStatus": "Active" } } } }</pre>
--	--

Calling an Area Scene

To activate an area scene, send a CreateRequest using commandprocessor on the area in which you intend to activate the scene. The body of the request should include the href of the scene you are activating, per the below example.

<pre>{ "CommuniqueType": "CreateRequest", "Header": { "URL": "/area/616/commandprocessor" }, "Body": { "Command": { "CommandType": "GoToScene", "GoToSceneParameters": { "CurrentScene": { "href": "/areascene/622" } } } } }</pre>	<pre>{ "CommuniqueType": "CreateResponse", "Header": { "StatusCode": "204 NoContent", "Url": "/area/616/commandprocessor" } }</pre>
---	---

Monitoring Resources

In addition to direct control of system resources, it is possible to monitor changes to these resources using either a ReadRequest or a SubscribeRequest. Using ReadRequest, you can pull the current state of the resource. Using SubscribeRequest, the client will receive a synchronous request from the processor confirming the subscription, and whenever the subscribed resource changes state, the client will also receive an asynchronous response from the processor detailing the change.

Monitor Zone Changes

The below code shows a ReadRequest on a dimmed zone resource. The response shows the current intensity level of the zone. This same code can be used for shade zones and HVAC zones.

<pre>{ "CommuniqueType": "ReadRequest", "Header": { "URL": "/zone/1698/status" } }</pre>	<pre>{ "CommuniqueType": "ReadResponse", "Header": { "MessageBodyType": "OneZoneStatus", "StatusCode": "200 OK", "Url": "/zone/1698/status" }, "Body": { "Intensity": 50 } }</pre>
--	--

```
"ZoneStatus": {  
    "href": "/zone/1698/status",  
    "Level": 75,  
    "Zone": {  
        "href": "/zone/1698"  
    },  
    "StatusAccuracy": "Good"  
}  
}  
}
```

To subscribe to zone-level changes in the Lutron system, the below command is used. At this time, it is not possible to subscribe to individual zones. The below will generate asynchronous responses any time any zone on the system changes.

```
{  
    "CommuniqueteType": "SubscribeRequest",  
    "Header": {  
        "URL": "/zone/status",  
        "Directives": {  
            "SuppressMessageBody": true  
        }  
    }  
}  
  
{  
    "CommuniqueteType": "SubscribeResponse",  
    "Header": {  
        "StatusCode": "204 NoContent",  
        "Url": "/zone/status",  
        "Directives": {  
            "SuppressMessageBody": true  
        }  
    }  
}  
  
{  
    "CommuniqueteType": "ReadResponse",  
    "Header": {  
        "MessageBodyType": "MultipleZoneStatus",  
        "StatusCode": "200 OK",  
        "Url": "/zone/status"  
    },  
    "Body": {  
        "ZoneStatuses": [  
            {  
                "href": "/zone/1698/status",  
                "Level": 50,  
                "Zone": {  
                    "href": "/zone/1698"  
                }  
            }  
        ]  
    }  
}
```

```
        "StatusAccuracy": "Good"
    }
]
}
}
```

Monitor Area Changes

The status of an area can be read through LEAP as well, using the same call used on zones. In the URL of the code header, call `/status` on a specific area `href` to read current information about that area.

{	{
"CommuniqueType": "ReadRequest",	"CommuniqueType": "ReadResponse",
"Header": {	"Header": {
"URL": "/area/616/status"	"MessageBodyType": "OneAreaStatus",
}	"StatusCode": "200 OK",
}	"Url": "/area/616/status"
	},
	"Body": {
	"AreaStatus": {
	"href": "/area/616/status",
	"Level": 75,
	"OccupancyStatus": "Occupied",
	"CurrentScene": {
	"href": "/areascene/622"
	}
	}
	}
	}

Refer to the complete API Protocol Guide for the relevant system for more information about the resources that can be monitored and/or subscribed to.

Next Steps

With the completion of Part 3 of the LEAP Quick Start Guide, you are encouraged to fully develop your integration using the complete API Protocol Guide for the relevant system. The full guide will provide more specific information about the commands and resources available to you.

Revision History

Revision	Released By	Release Date	Notes
A	Ctoth	August 2023	Initial Release